第2章

失敗しない構成管理ツールの導入法

――目的を持ってツールを活用しよう

田中友子

ソフトウェア構成管理の導入を考えたとき，多くの場合は市販ツールの導入を検討することになる．しかし，ツールを導入することそのものが目的になってしまい，何のために構成管理を行うのかを見失ってしまうことも珍しくない．構成管理ツールを効果的に導入・活用するためには，構成管理を導入する目的を明確にして，自分のプロジェクトに必要な機能を備えたツールを選ぶことが肝要である．　（編集部）

ソフトウェア構成管理（以下，構成管理）は，「構成管理ツール」と呼ばれる市販ツールを使わないと実施できないわけではありません．しかし，ツールを利用することにより，管理情報を入力しやすくなり，作業者により確実に入力してもらえる，などのメリットがあります．具体的には，以下の2点のような特徴を持ちます．

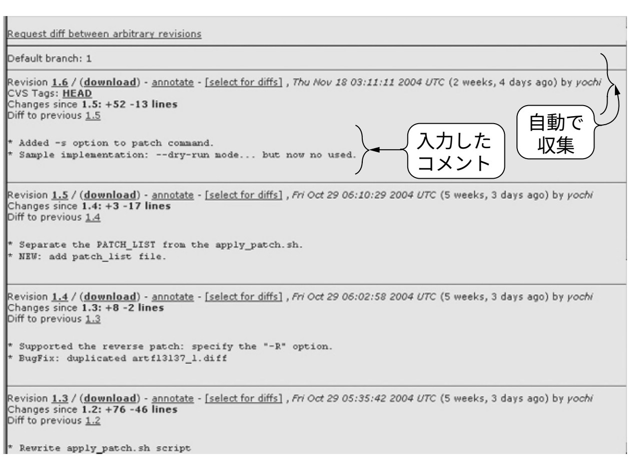


図1　構成管理ツールは入力作業をサポートする

CVSで登録した内容をWebブラウザから参照するツール「CVSWeb」の表示例．登録時はCVSのコマンドcvs commitを実行し，コメントを入力するだけである．つまり，ここに表示されているうちのコメント以外の項目（変更されたファイル名，変更日時，バージョン，変更ユーザ名）はすべて自動で収集されたものである．

1）管理情報を入力（収集）しやすくする

まじめに構成管理を行おうと思ったら，「いつ，だれが，どのファイルを，どういう目的で」変更したのかをいちいち記録する必要があります．変更記録用紙を渡されて，変更のたびに「日時，氏名，ファイル名，変更箇所」を書くことを想像してみてください．ファイル名はまちがえやすいし，日時は適当になりがちです．変更箇所は全部網羅されておらず，そのうえ時間がかかるし，めんどうです．

構成管理ツールを使えば，コードを変更した後で「チェックイン」という登録操作を行うことにより，操作を行ったユーザ名，ファイル名，日時が正確に記録されます．変更理由などの入力もできるだけ簡単に行えるようなくふうがなされています（**図1**）．また，ただ記録されるだけでなく，「一つ前のリリースを構成するファイル群」などといった特定の意味を持つファイル群を取り出したり，あるファイルの変更と合わせて変更されたファイルを探し出せるなど，入力した（収集された）記録も活用できます．

2）派生したソフトウェアを管理できる

ソフトウェア開発では，一つのソフトウェアをもとにして，それぞれ少しずつ異なる機能を実装した複数のソフトウェアを分岐して作り出していきます．例えば，ソフトウェアを複数のOSに対応させたり，顧客に合わせてカスタマイズを行ったりします（**図2**）．このような場合にめんどうなのが，各分岐先に問題（不ぐあいや修正箇所）があった場合の対応の管理です．その問題は共通の問題なのか，それとも個別の問題なのか，どの分岐に対して対応を行ったのか確認しなくてはいけません．すぐに確認できる開発環境とそうでない開発環境では，当然ながら，一つの問題処理にかかるコストが異なります．

世の中にはいろいろな構成管理ツールがありますが，そ

表1
なぜ作業に時間がかかっているのか？
問題がある作業をピックアップし，なぜ時間がかかっているかの理由と，入手や抽出に時間がかかっている情報を整理する．

問題がある作業	考えられる理由	入手や抽出に時間がかかっている情報
リリース時に修正された不ぐあいの一覧作成（いつも時間がかかる）	● 不ぐあいを修正したときのファイル名に漏れがある（あるいはまちがっている） ● ファイルのどこが修正の対象になったのかわからない	● 不ぐあいとファイル修正箇所との正確な対応づけ ● 前リリース時に使われたファイル群と今回のファイル群の差分
過去に行ったことがある作業（過去と同じだけの時間がかかる）	過去の作業時の記録が不明確	過去の作業の対象となったファイルとその箇所

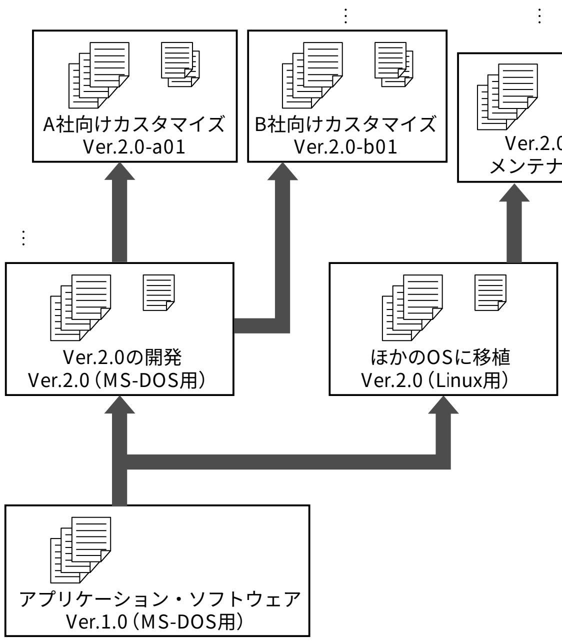


図2　ソフトウェアは次々と分岐していく
ソフトウェアを複数のOSに対応させたり，顧客に合わせてカスタマイズを行ったりするたびに，ソフトウェアは分岐していく．

れぞれ，ソフトウェアの分岐を効率的に管理するためのしくみを備えています．分岐が行われた基点となるソフトウェアや，分岐前，分岐後のすべての履歴が管理されるほか，分岐後の複数の変更について別の分岐へ反映させたり，反映を元に戻したりという操作を行えます．

開発者は，構成管理ツールから各分岐の最新のファイル群を自分のワーク・スペースに取り込み，変更作業を行います．A社向けのワーク・スペースとB社向けのワーク・スペースを切り替えながら変更を加えたり，ほかの開発者の作業状況を確認しながら変更を反映したりします．

このように便利な構成管理ツールなのですが，すべてのユーザがツールを使いこなせているわけではありません．そこで，どのような点に気をつければうまくツールを導入し，運用できるのかを，構成管理ツールの導入を支援してきた筆者の立場からお話ししたいと思います．

1　構成管理ツールを導入するにあたって

ツールを導入するにあたっていちばんたいせつなことは，何のために導入するのか，という目的をしっかりと持つことです．また，ISO9000やCMM（capability maturity model）などの取得（つまり，構成管理の導入そのもの）を目的とするのもいいのですが，「現実に起こっている，または潜在的に存在する問題を改善する」ことを第一の目的とするほうが，より効果的に導入できます．

● 問題点を洗い出す

まず，「効率が悪い/予定どおりに進まない」といったたぐいの問題がある作業をピックアップしてみましょう．ここで，「問題はとくにありません」という意見もよく出てくるのですが，「もっと簡単に終わるはずだったのに…」，「いつもこの作業は時間がかかるから，めんどうなんだよね」と思うことはありますよね？ このような作業が見つかったら，なぜ，その作業に時間がかかっているのかを考えます．すなわち，その作業において入手や抽出に時間がかかっている情報は何なのかを整理します（**表1**）．

ここまで整理すると，あなたのプロジェクトは構成管理ツールに何を求めているのかが見えてきます．ツールのデモンストレーションを見ているときやツールのトレーニングを受けているときも，つねに「問題となっている情報を効率良く収集するには，どの機能をどういうふうに使えばよいのか？」を考えます．これによって，自分のプロジェクトに合わせてより効率的にツールを運用できるようになります．

● ツールはあくまでも道具にすぎない

結局のところ，構成管理ツールを導入すれば問題がすべて解決する，というものではありません．あくまでもツールは，開発環境を改善する能力を備えている，単なる道具



図3　SourceForge.jpのWebサイト

SourceForge.jpは，オープン・ソース・ソフトウェアの開発者に対して，Web上にプロジェクト管理環境を無償で提供するWebサイトである．URLは「http://sourceforge.jp/」．

の一つにすぎないのです．道具が効果を発揮できるかどうかは，使いかたによります．

ところで，「SourceForge.net」や「SourceForge.jp（**図3**）」というWebサイトをご存じでしょうか？これは，オープン・ソース・ソフトウェアの開発環境を提供しているWebサイトです．バージョン管理ツールであるCVSやバグ追跡ツールなどといった構成管理に役立つ環境を，オープン・ソース・ソフトウェアの各開発プロジェクトに提供しています．

現時点で1,000以上のプロジェクトがここで運用されていますが，おもしろいのはその利用方法の違いです．CVSだけ，メーリング・リストだけなど，一部の機能だけを使っているプロジェクトもあれば，すべての機能をフル活用しているところもあります．傾向としては，エンド・ユーザの多いプロジェクトのほうが，メーリング・リストよりもバグ追跡ツールやリリース管理ツールをじょうずに活用し，エンド・ユーザと開発者の間のコミュニケーションを円滑に行っているようです．

● 内製ツール vs. 市販ツール

市販の構成管理ツールの価格は，1ユーザ当たり10万～50万円です．これはソフトウェア開発ツールなどと比べると高価です．そこで，「市販ツールを購入せずに，内製することに決めました」という話を聞くことがあります．社内の標準に適合するようにツールを内製できれば，その分ユーザの負担は減りますから，成功すればすばらしいことです．ただ，内製するとコストが抑えられるかというと，かならずしもそうとは言えません．実際，内製に挑戦したけれども途中で挫折し，市販ツールを再検討することにした，という話も何度となく耳にしています．

市販ツールは多くの開発プロジェクトに導入され，そのフィードバックを受けてノウハウが盛り込まれています．各開発チームのスタイルに合うように設定を変更したり，カスタマイズしたりすることも簡単に行えます．筆者は，内製にこだわって導入が遅くなるよりは，とりあえず市販ツールで始めてみたほうがよいと考えます．

2 ツール・ベンダを利用する

構成管理ツールは購入しやすいツールではありませんし，購入してすぐに使い始められるツールでもありません．導入前から運用開始後までツール・ベンダまたは販売代理店に協力してもらう必要があります．

● ツールやベンダを比較検討しよう

最近は，米国で開発された主要なソフトウェア構成管理ツールが，かなり日本で販売されるようになってきました．そうなると，その中から自分のプロジェクト環境に合ったツールを選ばなくてはなりません．まずは，前述のように目的を明確にしたうえで，各ツールの調査を始めることになります（話を聞くよりは，実プロジェクトで実際にそのツールを使ってみるのがいちばん良い．ただし，いくつかのツールを比較したいとき，いちいち使って試すのは時間がかかりすぎる）．

一案としては，目的に沿った要求事項を自分で表などにまとめ，各ツール・ベンダにその回答を埋めてもらうとよいでしょう．各ツールの能力だけでなく，ツール・ベンダの技術力や対応のよしあしがわかります．別案としては，CVSやSubversion注1など，入手しやすくとっつきやすいオープン・ソースのツールを使ってみるのもよいでしょう．合えばそれでよいですし，合わなければツール・ベンダにCVSやSubversionの問題点を提示して，そのベンダが提供するツールによって問題を解決できるかどうかを問い合

注1：Subversionとは，オープン・ソースのバージョン管理システムである．「http://subversion.tigris.org/」などから各OS用のバイナリ・ファイルを入手できる．

わせてみます．ほとんどのツールがCVSやSubversionとの比較表を持っています．また，それらのデータをツールに移行することもできるはずです．

ちょっとおうちゃくして，各ツール・ベンダに他社のツールとの比較表を求めることもできそうですが，これはお勧めしません．ツール・ベンダは自社のツールの欠点は必要なければあえて明示しません．ツール・ベンダ間では競合製品の情報を入手することは意外と難しいからです．

● ツールの運用を開始する

実際に構成管理ツールを導入するとき，ユーザとなる開発者の負担ができるだけ軽くなるようなくふうが必要です．例えば，チェックインのときにコメントをできるだけ簡単に入力できるようなフォーマットを作ったり，構成管理ツールを導入すれば不要になるはずの報告や会議をできるだけ減らす，などの対処を行います（ツールを熟知しているツール・ベンダはこのようなアイデアをいろいろ持っているはずなので，ぜひ聞いてみてほしい）．

運用を始める前に，まず，開発時にかならず収集したい情報を明確にします．そして，その情報を集めるには，日ごろから開発者にどのような操作（情報の入力）を行ってほしいかを考えます．この段階でも，ツール・ベンダに相談してみましょう．ツール・ベンダは同じような質問と回答例（解決策）をノウハウとして持っています．また，よく要望のある自動化スクリプトなどは，簡単なものであれば用意されているはずです（例えば，チェックイン時にメールを送信するようなものなど）．ある程度範囲が絞られており，ポイントが明確になっていれば，サポートする側も的確に答えやすくなります．こうして，どんどんツール・ベンダのノウハウを引き出しましょう（**表2**）．

開発者に求める操作が明確になったら，開発者向けの簡単なマニュアルを作ってトレーニングを行い，いよいよ運用を開始します．

なお，どのような情報を収集するのかを決めることから運用開始までの全ステップを，ツール・ベンダにコンサルティングとして依頼することも，もちろんできます（その分，費用はかかるが…）．

● どんどん質問して，ツールをよく知ろう

準備を万全にして運用を開始しても，ユーザはそれなりにいろいろな使いかたをします．そして，質問や問題の報告がツール導入担当者にどんどん送られてきます．こういうときは，どんどんツール・ベンダのサポート窓口にそのまま送ってしまいましょう．サポート窓口と多くやりとりするほど，そのツールに対する知識が深まります．ツール・ベンダのほうでも，質問の量と内容で，顧客がツールを使いこなしているかどうかを判断できます．

質問するときは，行いたいことが明確になっているほうが，より適切な回答を受け取ることができます．例えば，単に「モジュールの追加方法を教えてください」と質問すると，「マニュアルのどこそこを見てください」という回答が返ってきてしまいます．これに対して，「モジュール○×用のライブラリを新しく追加することになったのだけれど，これは別のモジュールでも参照する予定で，配置場所やアクセス権をどう設定するかで悩んでいます」と質問すれば，ツール・ベンダは最善だと思われる方法を提案してくれます．

構成管理ツールをじょうずに利用して，スムーズな開発を行いたい（行ってほしい），むだな作業を減らしたい（減らしてほしい）という思いは，質問する側，質問を受ける側とも共通のものです．なので，単にユーザとベンダという関係の質問のやりとりではなく，より深いコミュニケーションをとって，お互いの環境への理解を深めていこうとする姿勢がたいせつです．このことは構成管理ツールの運用にもプラスに働くでしょう．

表2　ツール運用に関する提案例

ツールを「こう使いましょう」と決めてもなかなか全員に浸透させるのは難しい．そこで，ツールの機能やインターフェースを利用して，作業を自動化したり，作業を強制したりするとスムースに運用しやすくなる．

提案例	メリット
定期的に特定のラベルを付けたファイルを使ってビルドを行い，でき上がったものを自動的にチェックインできるように設定する	ビルド作業を自動化できる
（テスト期間中やリリース後の保守フェーズなどにおいて）チェックイン時にバグ番号を入力しないとチェックインできないようにする	● バグ番号の入力漏れがなくなる ● バグの対処以外の変更を認めない（変更したかったらバグ登録を行うように徹底させる）ようにすることができる

たなか・ともこ
VA Linux Systems Japan（株）

<筆者プロフィール>
田中友子．構成管理ツールや開発コラボレーション・ツールの国際化およびユーザ・サポートに従事．いつも日本独自の問題（企業文化や文字コードの問題など）を海外のベンダに理解してもらうために四苦八苦しています．